APPENDIX I
Marked Version of Specification Pages

Page 6, lines 14-16

Figures [5a and 5b] 5a, 5b, and 5 c illustrate a detailed view of a debouncing logic circuit and a storage circuit for storing states on signal lines according to a preferred embodiment of the present invention;

Page 6, lines 17-19

Figures [6a and 6b] 6a, 6b, and 6c illustrate a detailed view of a debouncing logic circuit and a remote wakeup circuit containing a plurality of debouncing logic circuits according to a preferred embodiment of the present invention;

Page 7, lines 4-5

Figures [8a and 8b] 8a–8d illustrate a bus monitor and remote wakeup unit state machine according to a preferred embodiment of the present invention.

Page 15 line 11 through Page 16 line 7

Referring now to Figures 5a and 5b, a block diagram illustrates a portion of the bus monitor unit 340 of the PIE 320 responsible for generating debounced signal flag lines for the various states of the LINE STATE signal line 410 according to a preferred embodiment of the present invention. As discussed previously, the two bits of the LINE STATE signal line 410 can represent a total of four distinct states: SE0, J state, K state, and SE1. According to the USB version 2.0 technical standard, state SE1 is reserved for testing purposes and is not used during normal operations. The schematic diagram displays three debouncing circuits 505, each driven by a line representing an output from a data register storing the state on the LINE STATE signal line 410. Notice that there is one debouncing circuit 505 for each usable state of the LINE STATE

signal line 410. For example, if the LINE STATE signal line 410 were carrying the state J state, then the data register for the J state would be containing a binary one value while the registers for the SE0 and K states would be containing a binary zero value. Notice that the actual contents of the registers could vary depending on the logical convention used, i.e., logic low true (a zero represents a true value) or logic high true (a one represents a true value).

Page 16 line 8 through Page 17 line 3

Referring now to Figure 5[b]c, a block diagram illustrates the storing of the state of the LINE STATE signal line 410 in data registers according to a preferred embodiment of the present invention. The state of the LINE STATE signal line 410 are stored in register pairs 542, of which, there are three pairs, one for each usable state of the LINE STATE signal line 410. The register pair 542 comprising a cascade of D-type flip-flops 546 and 547, which are clocked by the clock signal as generated by the UTM 310. The cascade of flip-flops 546 and 547 are used to synchronize LINE STATE signal line 410 to the system clock: CLK. Even though LINE STATE signal line 410 is synchronous to the system clock, but due to clock skew, present between UTM 310 and PIE 320, the signal on LINE STATE signal line 410 input to the bus monitor unit 340 may not satisfy the required setup time requirements, so the cascade of flip-flops 546 and 547 are used to synchronize the signal on the LINE STATE signal line 410 with the system clock. In situations when the UTM 310 is implemented as a discrete component and is separate from the PIE 320, the clock skew between the signal on the LINE STATE signal line 410 and the system clock of the PIE 320 can be relatively large.

APPENDIX I
Marked Version of Specification Pages

Page 17 line 15 through Page 18 line 10

Referring back to Figures 5a and 5b, the debouncing circuit 505 comprising a cascade of three D-type flip-flops 510, 512, and 514 (as discussed previously, other types of flip-flops are usable with the addition of a small amount of glue logic), two multiplexors 516 and 518, and a logical AND gate 519. As discussed previously, input into the debouncing circuit 505 is the output of a data register used to store the state of the LINE STATE signal line 410. The output becomes the input to the first D-type flip-flop 510. The input of the second and third D-type flip-flops 512 and 514 are multiplexed between the output of the previous flip-flop and the output of the storage register with the output of the storage register also being used as the select line for the multiplexor. When the output of the storage register is zero, i.e., when the state of the LINE STATE signal line 410 is not the state driving the particular debouncing circuit 505, the flip-flops in the debouncing circuit 505 are set to zero. When the output of the storage register is one, i.e., when the state of the LINE STATE signal line 410 is the state driving the particular debouncing circuit 505, the input of a flip-flop is the output of the previous flip-flop or the output of the storage register if the flip-flop is the first flip-flop.

Page 21, lines 3-15

Similar to the debouncing circuit 505 discussed in Figures 5a and 5b, the debouncing circuit 600 comprises a flip-flop cascade 601 and combinatorial logic. As is the case previously, D-type flip-flops are preferred, although it is possible to use other types of flip-flops with the addition of a small amount of glue logic. According to a preferred embodiment of the present invention, there are

a total of six D-type flip-flops in the debouncing circuit flip-flop cascade 601. Input into a first flip-flop 602 is the signal line being debounced. The first flip-flop 602 is clocked by the clock signal generated by the UTM 310. This same clock is also used to clock the remaining five flip-flops in the cascade 601. The output of the first flip-flop 602 is the input into a second flip-flop 604. The first two flip-flops 602 and 604 serve synchronization function (removal of any clock skew) that is similar to the function provided by the data registers 542 from Figure 5c[b].

Page 24 line 14 through Page 25 line 4

Referring now to Figures 6b and 6c, a block diagram illustrates a remote wakeup circuit 650 with debouncing circuits 600 for signal pins according to a preferred embodiment of the present invention. According to a preferred embodiment of the present invention, the remote wakeup circuit 650 as displayed in Figures 6b and 6c is for a USB removable storage device with the capability of accepting data cartridges and solid-state memory devices. The remote wakeup circuit 650 can be used with other peripherals with and without minor modifications, depending on the needs of the peripheral, for example, a different number of signal pins, etc. In the removable storage device application, the remote wakeup circuit 650 can be used to detect when a user has inserted or ejected a data cartridge or a solid-state memory device, for example.

Page 29 lines 5 - 19

Referring now to Figures 8a-8d (referred to collectively as Figure 8) a flow diagram illustrates a combination flow-chart and state machine 800 for a bus monitor 340 according to a preferred embodiment of the present invention. According to a preferred embodiment of the

present invention, the state machine 800 executes in the bus monitor 340 and uses as inputs debounced and stable signals provided to the bus monitor 340 by various debounce circuits, discussed previously. The bus monitor 340, under control of the state machine 800, powers up in an idle state, POWER_UP_RESET (block 801). The bus monitor 340 will remain in the idle state until it receives a signal (Usb_cnt) from the MCU 330 that a peripheral has been connected to the USB and is ready to communicate with host or hub. When the bus monitor 340 is in the POWER_UP_RESET state and the MCU 330 asserts Usb_cnt = 1, the bus monitor 340 remains in the POWER_UP_RESET state (block 802) for a preferred time period equal to 125 micro-seconds in full speed mode to permit the signal LINE STATE time to settle.

Page 29 line 20 through Page 30 line 9

After waiting for a period of 125 micro-seconds in block 802, the bus monitor 340 enters a CONNECT_FS state and continues its power-up routine by staying in full speed mode (block 804). It is preferred that USB devices initially power-up in full speed mode and then perform necessary handshaking to determine actual connection speed. In block 806, the bus monitor 340 checks to see if a bus reset has occurred. This is performed by checking if the signal (line_st_se0_db) is equal to 1, if it has, then LINE STATE has been continuously SE0 for preferably four to six micro-seconds, meaning that there is a bus reset request from the hub. If a bus reset has occurred, the bus monitor 340 enters state HS_DET_HANDSHAKE1 and jumps to block 850 [(Figure 8b)], which will be discussed below.

Page 31, lines 10-17

If the bus is active and has completed the first bus reset after power up, the bus monitor 340 jumps to block 848, where the bus monitor 340 checks if high speed mode is enabled, this checks if high speed mode has been enabled prior to entering suspend mode. If high speed mode is enabled, the bus monitor 340 enters state HS_BUS_ACTIVE and jumps to block 874 [(Figure 8b)], which will be discussed below. If high speed mode is not enabled, the bus monitor 340 enters state FS_BUS_ACTIVE and jumps to block 890, which will also be discussed below.

Page 32, lines 3-12

If bus_wakeup is equal to one (block 820), then the bus monitor 340 enters state CHK_RESUME_GLITCH and block 838 and checks if a bus reset has occurred or if the host has resumed signaling. In block 840, the bus monitor 340 checks if the bus is idle. If the bus is idle, then the bus monitor 340 returns to the SUSPEND state and block 818. If the bus is idle, then the bus monitor 340 decides that the bus_wakeup was triggered by a glitch in the LINE STATE signal line 410. If the bus is not idle, then the bus monitor 340 checks if a bus reset has occurred (block 842). If a bus reset has occurred, the bus monitor 340 enters state HS_DET_HANDSHAKE1 and jumps to block 850 [(Figure 8b)], which will be discussed below.

Page 34, lines 8-16

The bus monitor 340 then enters state WAIT_RESUME_END and block 832, where the bus monitor 340 waits for the resume to end (when signal full_speed_eop_db = 1) (block 834). Once the resume ends, the peripheral returns to its previous speed mode by checking if high_speed_mode = 1 (block 836). If high_speed_mode is equal to one, then the bus monitor 340

enters state HS_BUS_ACTIVE and jumps to block 874 [(Figure 8b)], which will be discussed below. If high_speed_mode is not equal to one, then the bus monitor 340 enters state FS_BUS_ACTIVE and jumps to block 890 [(Figure 8b)], which will be discussed below.

Page 34 line 17 through Page 34 line 6

[Referring now to Figure 8b, a flow diagram illustrates a continuation of the flow diagram displayed in Figure 8a according to a preferred embodiment of the present invention.] Turning now to state HS_DET_HANDSHAKE1 and block 850, the bus monitor 340 sets the high speed transceiver select, full speed termination select, disable bit stuffing, and non-return to zero signaling and enables transmission of chirp K for one millisecond. This signals to the host (or hub) that the peripheral is high speed capable. The bus monitor 340 then enters state HS_DET_HANDSHAKE2 and block 852, where it holds the signals set in block 850 (Figure 8C) for a full four micro-seconds to ensure data in the UTM 310 data registers is transmitted to the bus with bit stuffing disabled and NRZI encoding mode, so host will observe chirp K through the whole duration.